MANCHESTER
1824
The University of Manchester

cpu gpu
DualSPHysics

Investigating Energy Consumption of SPH Simulations Using DualSPHysics

Amr Mohamed, Benedict D. Rogers

University of Manchester

Background and Motivation

- Scientists now rely on GPUs in HPC systems to conduct their simulations
- DualSPHysics is a key tool for modelling fluid behaviour using HPC systems
- Data-center and supercomputer energy use is expected to increase significantly in the coming years

- **Purpose of this work is therefore to answer the following questions:**

How much energy do SPH simulations consume?

What practical methods can reduce this energy use?

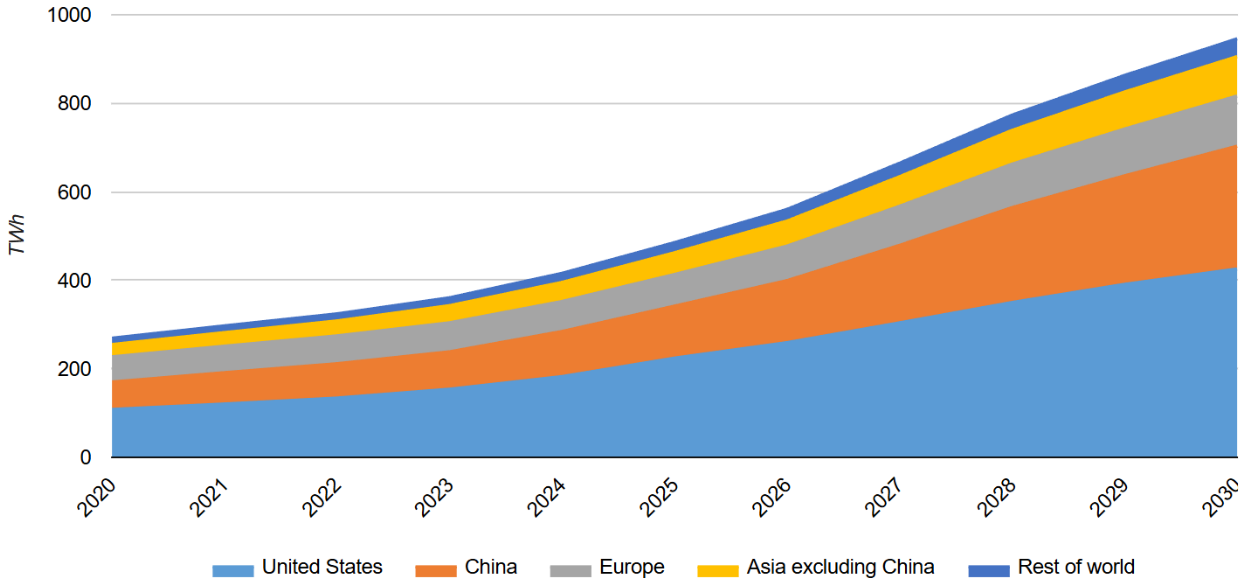


Figure 2: Data Centers Electricity Consumption Forecast 2020-2030

Source: IEA

Energy Consumption: How is it estimated?

Energy = Power × Time

Time-to-Solution (TTS) → taken from DualSPHysics .out file

Power → 3rd party tools measure power metrics of different hardware, averaged across simulation

Hardware power consumption depends on the **Setup**

Main hardware responsible → GPUs (Graphics Processing Units)

Note: It is **crucial** to **attempt** to measure power output of **all** components

Not always possible, depends on the **setup**

GPU remains the priority when it comes to measurements

Hardware Setup #1: Personal Laptop

- GPU: RTX 2060 MAX Q, CPU: Ryzen 9 4800HS

Data Collection:

- NVIDIA SMI for GPU metrics
- MSI Afterburner for system-wide metrics (incl. CPU)
- Total energy consumption roughly estimated through combining CPU and GPU power metrics
    - Fan cooling, display, other electronics have minimal power draw compared to CPU and GPU

Hardware Setup #2: University’s HPC system (CSF)

- Comprised of dozens of nodes
  - Node used:
    - 4x Nvidia A100 GPU,
    - 4x 48-Core AMD EPYC CPUs,
  - For simulation: 1 GPU, 8 CPU cores

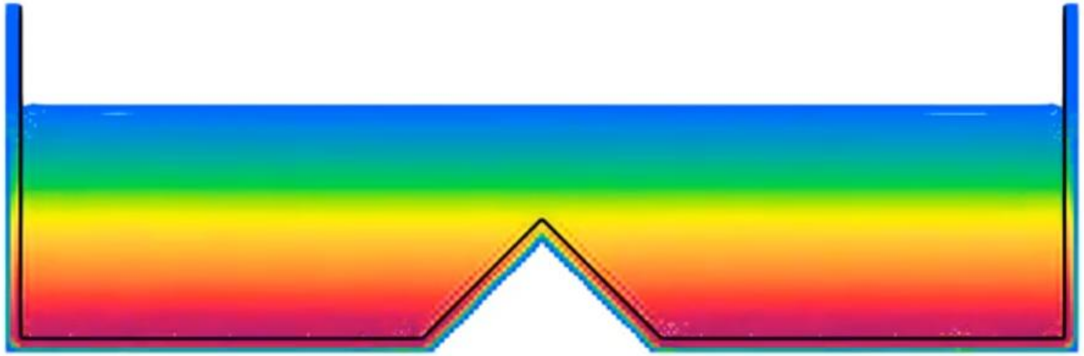
Compute nodes from the CSF
Source: Manchester IT services

Data collection

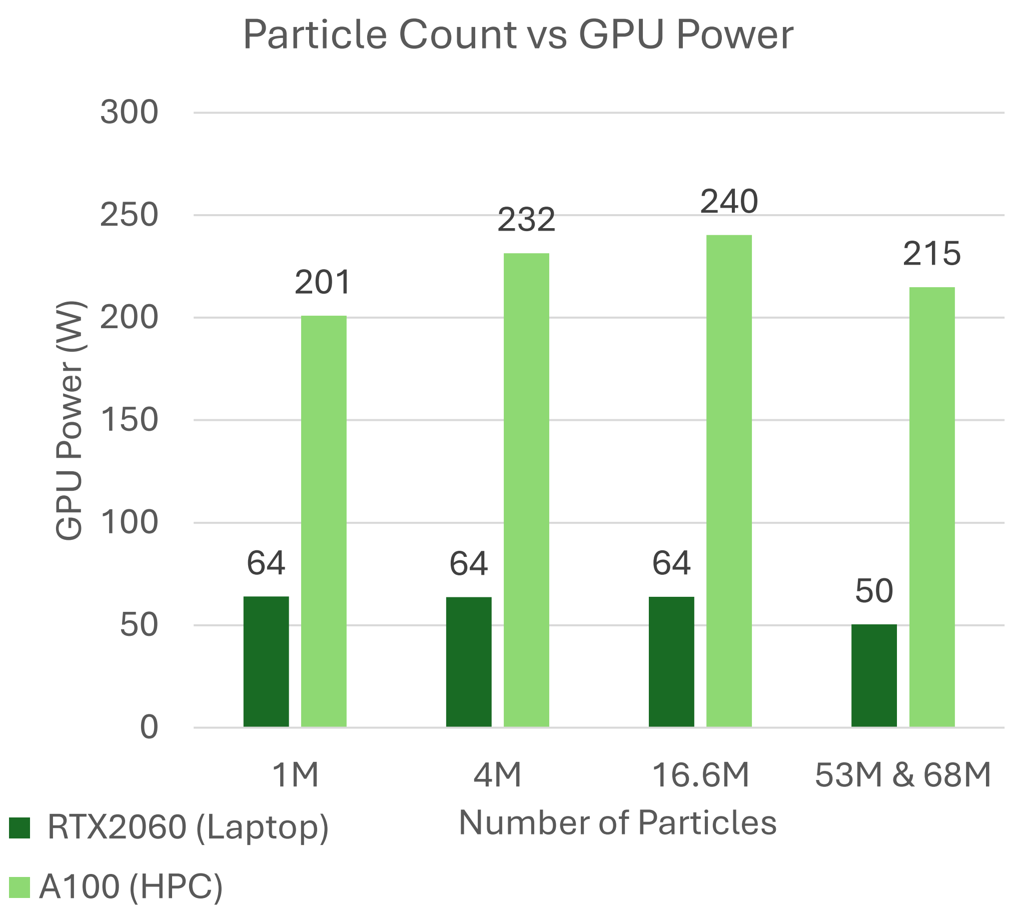
- Nvidia SMI for the single A100 GPU
- SLURM reports consumption of the entire node (through IPMI module)
  - Captures power consumption of entire board that the node operates on
  - Lacks granularity: Includes the idle components not being used

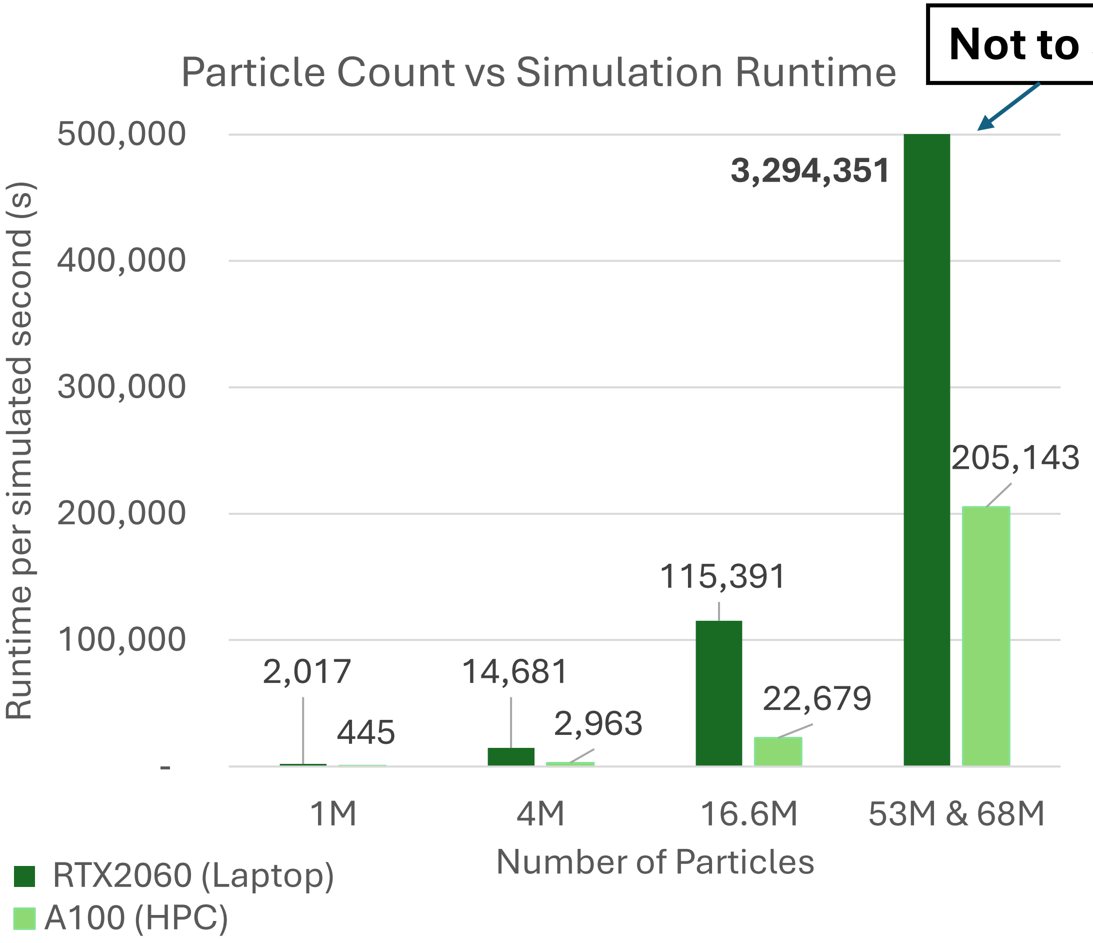
DualSPHysics Setup

- Static water simulation: StillWedge test case provided with DualSPHysics
- Time steps are consistent, time taken to for each timestep is the same
- Allows us to extrapolate data more predictably and accurately→ No need to run hours of simulations
- All metrics shown will reflect 1 simulated physical second



Results: Energy consumption on default settings

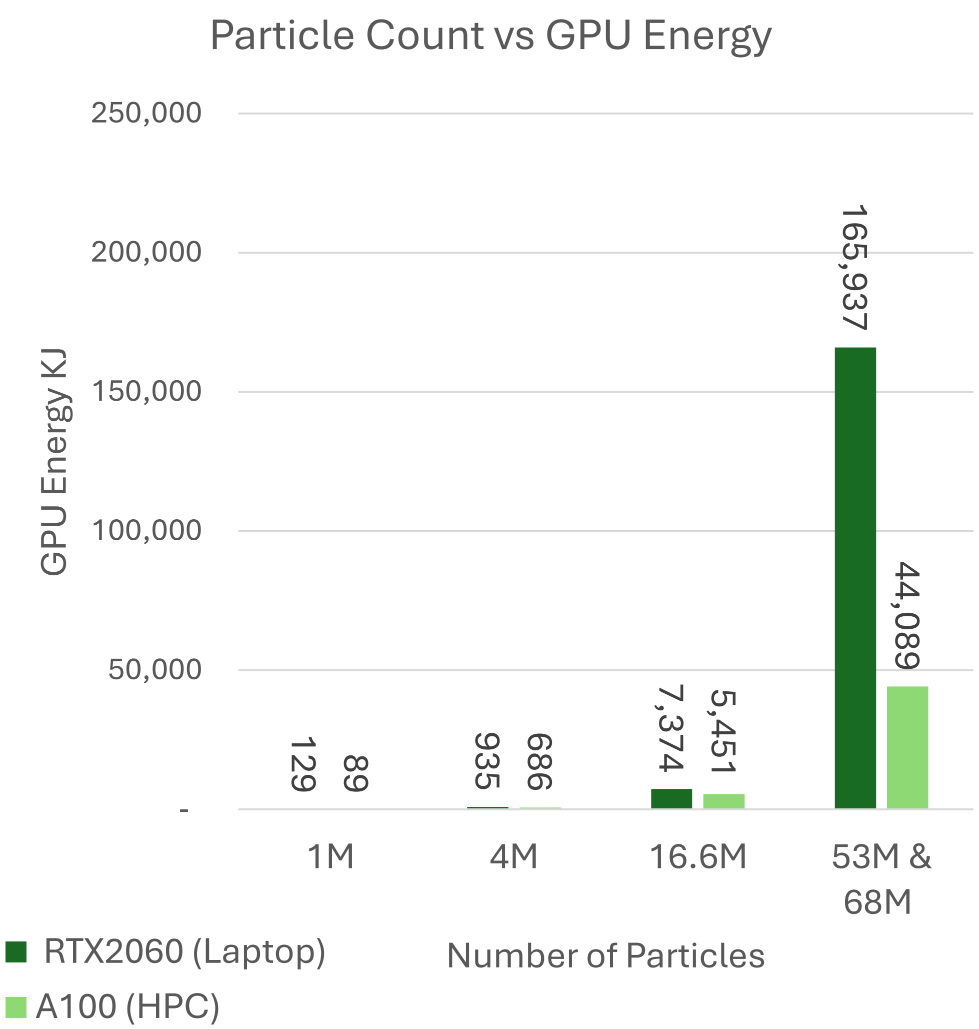




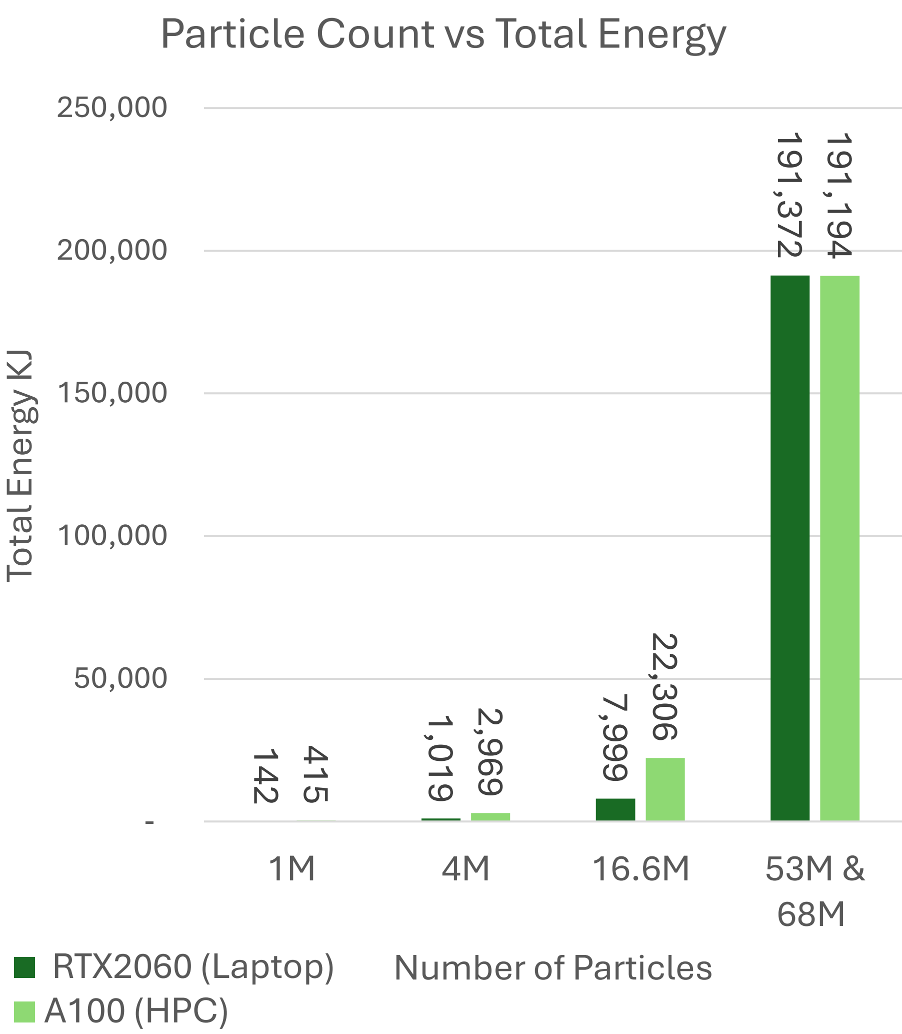
Not to scale

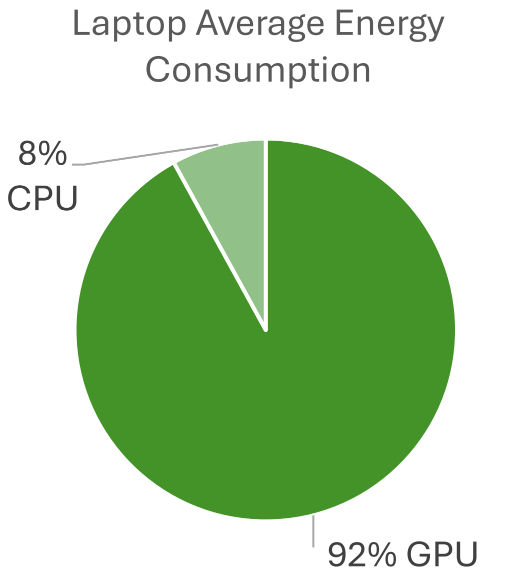
Modelling up to 68 Million Particles

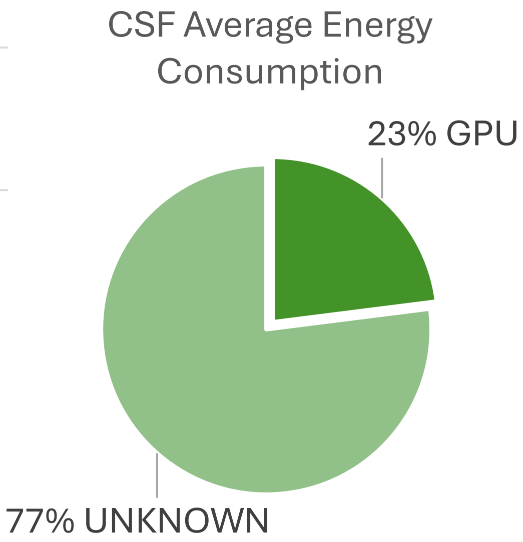
Results: Energy consumption (GPU vs Total)


Particle Count vs GPU Energy

Modelling up to 68 Million Particles


Particle Count vs Total Energy


Laptop Average Energy Consumption


CSF Average Energy Consumption

Results: Examples of Equivalent Consumptions

Equivalent Energy Consumption

1M Particles (Low-Res)	• Laptop: Boiling a quarter of a kettle (0.04kWh) • CSF: Boiling a full Kettle boil (0.12kWh)
16M Particles (Med-Res)	• Laptop: Running electric oven for 30 mins(2.2 kWh) • CSF: : Running electric oven for 90 mins (6.20 kWh)
53-68M Particles (High-Res)	• Both: Week of average UK household electricity use (53.1kWh)

What practical methods can reduce this energy use?

“Underclocking”

The practice of reducing the clock speed of a processing unit

How does Underclocking work?

All processing units, (CPUs, GPUs), are fundamentally complex arrangements of billions of transistors

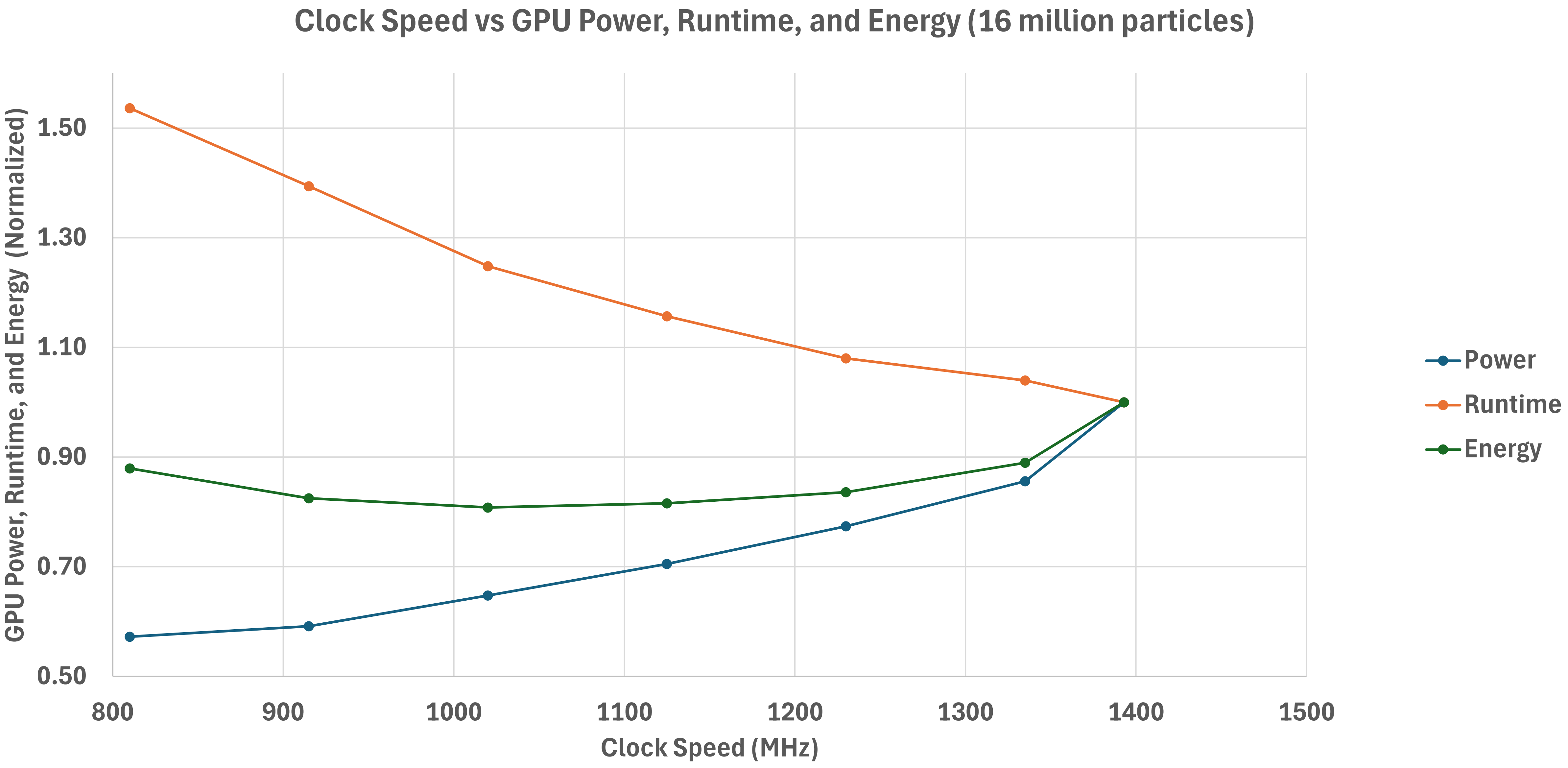
Transistors need to switch on and off extremely quickly to perform operations, such as calculating forces on particles

By reducing the speed of this switching, we can cause significant drops in power draw, at the expense of performance

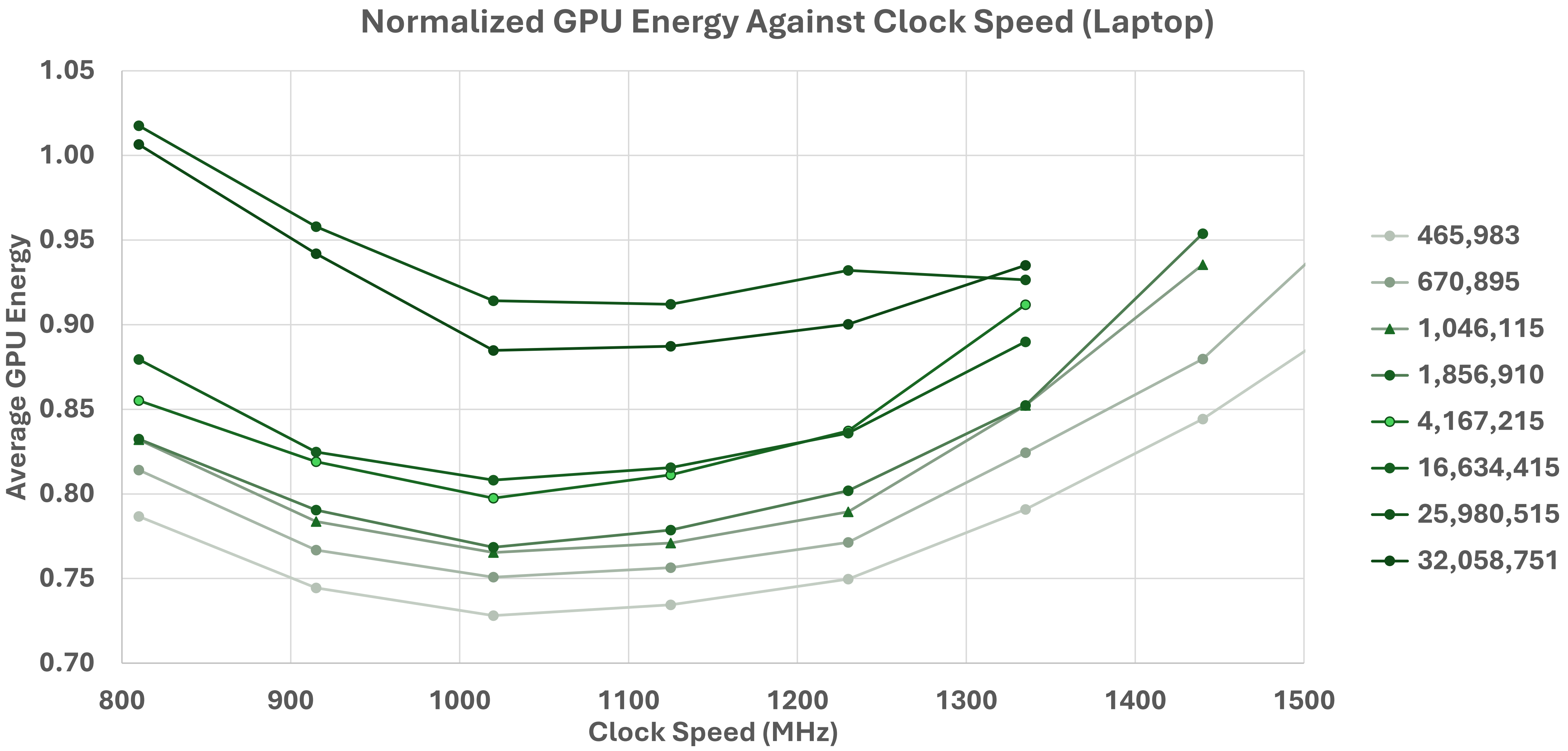
ENERGY = POWER * TIME

If the TTS increases dramatically, reducing the power no longer gives the intended benefit

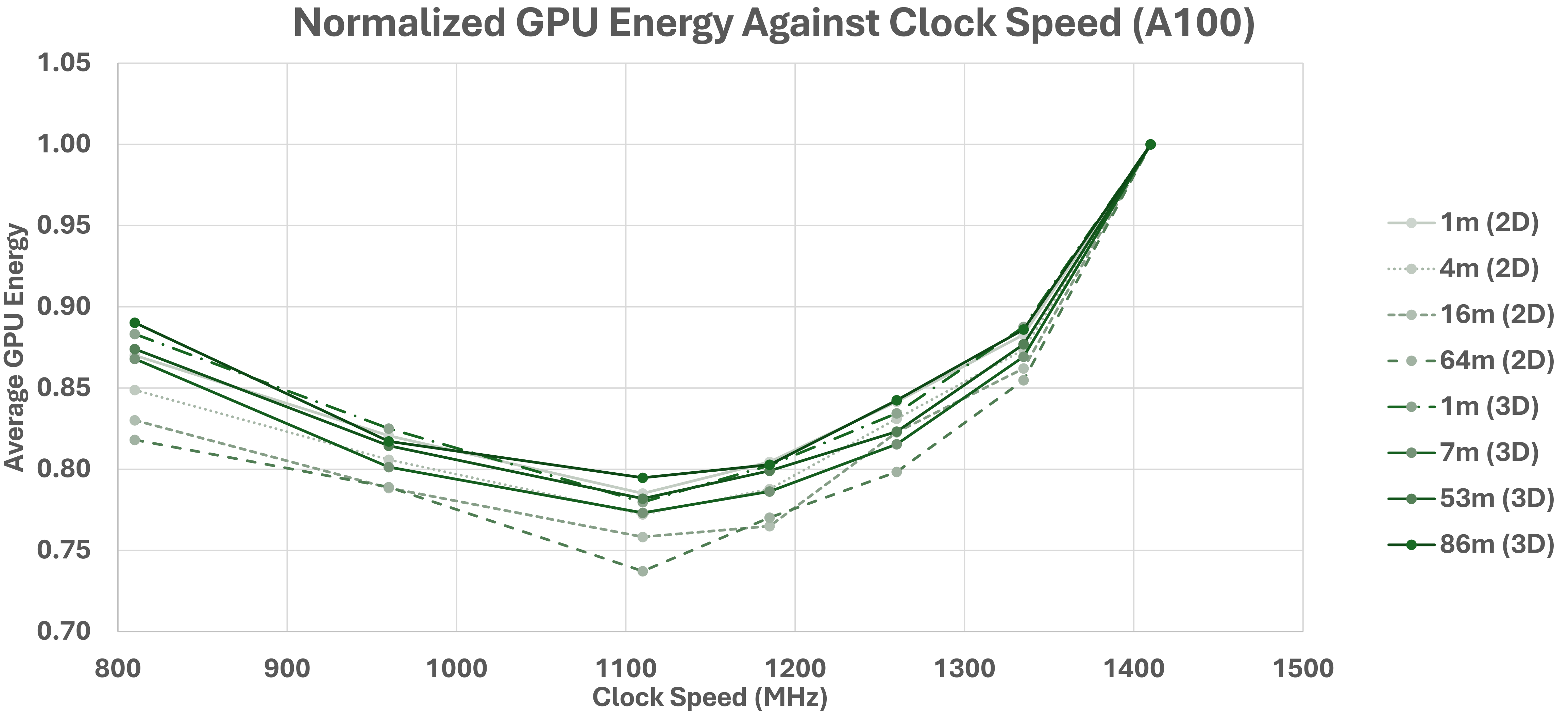
Underclocking Results



Underclocking Results



Underclocking Results



Underclocking Results: Optimal Frequencies

Heatmaps of Normalized Energy Consumption (for varying particle count)

Clock Speed	1m	2m	4m	16m	26m	32m
810	0.88	0.88	0.93	0.93	1.04	1.06
915	0.83	0.83	0.86	0.89	0.98	0.99
1020	0.80	0.81	0.84	0.85	0.93	0.93
1125	0.80	0.81	0.84	0.84	0.92	0.92
1230	**0.81**	**0.83**	**0.86**	**0.85**	**0.93**	**0.92**
1335	0.86	0.86	0.92	0.90	0.93	0.95

Clock Speed	1m (2D)	4m (2D)	16m (2D)	64m (2D)	1m (3D)	7m (3D)	53m (3D)	86m (3D)
810	1.33	1.25	1.24	1.18	1.31	1.35	1.34	1.27
960	1.18	1.14	1.12	1.12	1.25	1.18	1.20	1.20
1110	1.44	1.08	1.04	0.98	1.18	1.12	1.13	1.10
1185	1.17	0.98	1.00	1.02	1.08	1.05	1.07	1.04
1260	1.14	1.00	1.00	0.96	1.08	1.01	1.00	1.02
1335	1.06	0.91	0.88	0.88	0.91	0.91	0.91	0.94
1410	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Laptop

- Optimal frequency between around 1020-1125 MHz
- 12%-23% GPU energy savings
- 8%-20% Total energy savings

HPC

- Optimal frequency for GPU savings: 1110 MHz
  - 21-24% GPU Energy savings
- Optimal frequency for total energy savings: 1335 MHz
  - **6%-12% Total Energy savings**

Concluding Remarks

- Personal workstations equipped with mid to high-end GPUs are sufficient for running medium sized simulations
- Underclocking alone has a considerable impact on energy savings across the entire system despite the high overheads in HPC systems (6-12% energy reduction!)
- Further Work: Undervolting, implementation of workload-aware frequency scaling (DVFS)

Acknowledgments

- Special thanks **to Martin Wolstencroft and Christopher Grave** from the Research IT team for their assistance in the use of the Computational Shared Facility at The University of Manchester

- School of Engineering at the University of Manchester

MANCHESTER
1824
The University of Manchester

cpu gpu
DualSPHysics

Thank you for your time!

Amr Mohamed: amr.mohamed@student.manchester.ac.uk

Benedict D. Rogers: benedict.rogers@manchester.ac.uk